Serial No. 09/409,613
Docket No. RO999091
Firm No. 0021.0002

REMARKS

1. Claims 1-6, 10-15, 19-24, 28-36, 41, 43, and 45 are Patentable Over the Cited Art

The Examiner rejected claims 1-6, 10-15, 19-24, 28-36, 41, 43, and 45 as obvious (35 U.S.C. §103) over Waters (U.S. Patent No. 6,535,867). Applicants traverse for the following reasons.

Independent claims 1, 10, and 19 concern processing an input file in a file system, wherein the input file has an input file name, by: storing in cache memory a data structure generated by applying a function to all file names in a file system to determine values corresponding to the file names, wherein the data structure indicates those values corresponding to the file names to indicate all file names used in the file system; applying a function to map the input file name to a value; and scanning the data structure in cache memory without reading directory information from storage locations in a storage device to determine whether there is a preexisting file in the file system having a name that maps, according to the function, to the same value to which the input file name maps, wherein two files that map to a same value according to the function are capable of having a same name.

The Examiner cited col. 7, line 40-col. 8, line 14 and FIG. 7 of Waters as teaching the claim requirement of storing in cache memory a data structure generated by applying a function to all file names in a file system to determine values corresponding to the file names, wherein the data structure indicates those values corresponding to the file names to indicate all file names used in the file system. (Fifth Office Action, pg. 3). Applicants traverse.

The cited cols. 7-8 a hash function that is applied to requests for a file identified by a Uniform Resource Identifier (URI) on a web server. The hash value provides an index into the memory where the requested file is located in an external memory. Thus, the cited cols. 7-8 of Waters discusses how a hash may be applied to files requested on a web server that are maintained in an external memory, such as an EEPROM 320, shown in FIG. 7.

Nowhere does the cited Waters disclose the claimed data structure generated by applying the hash to file names. Waters does not generate a data structure for the hash values. Instead, the

hash values in Waters provides an offset into an EEPROM memory where the URI file that may be requested is stored. The memory of Waters does not teach or suggest a data structure generated by applying a function to all file names in the file system, but instead is a memory used to store files requested by GET requests to the web server.

Further, nowhere does the cited cols. 7-8 of Waters anywhere teach, suggest or mention the claim requirement of applying a function to all file names in a file system to determine values corresponding to the file names, wherein the data structure indicates those values corresponding to the file names to indicate all file names used in the file system. Instead, the hash of Waters is only intended to be used for files in a web server memory that may be requested using an HTTP GET request. Nowhere does the cited Waters anywhere suggest applying its hash to all files in a file system. Instead, the hash of Waters only applies to a limited set of files that may be requested form the web server, not all files in the web server file system.

In fact, Waters teaches away from this claim requirement because the hash of Waters is intended to provide an offset into an EEPROM 32K bytes in size. All the files in a file system of a computer could provide hash values far exceeding the number of entries in a memory device such as an EEPROM. Thus, Waters teaches away from a hash that is intended to apply to all files in a file system and produce that many values because the hash of Waters provides an offset into a memory device.

The Examiner further cited col. 8, lines 15-31 of Waters as teaching the claim requirements. (Fifth Office Action, pg. 4) The cited col. 8 mentions that the hash value is used as an index into the external memory, e.g., an EEPROM, to retrieve the file at that index location. The cited col. 8 further includes a verification system to ensure the chosen file is the correct file, and provide collision handling.

Although the cited Waters discusses applying a hash to a URI file name and dealing with collisions to avoid returning the incorrect file, nowhere does the cited Waters anywhere teach, suggest or mention applying a hash to all file names in a file system to determine values corresponding to the file names, wherein the data structure indicates those values corresponding to the file names to indicate all file names used in the file system. Instead, the cited Waters only

applies the hash to those files intended to be accessed in the Web Server using GET requests, not all the files in the web server file system as claimed.

Yet further, nowhere does the cited Waters anywhere teach or suggest the claim requirement of scanning the (hash) table to determine whether there is a preexisting file having the same name, such that two files that map to a same value according to the function are capable of having a same name. Instead, the cited Waters is concerned with checking whether the correct file is returned due to collisions. Nowhere does the cited Waters anywhere teach, suggest or mention using the hash to determine whether a preexisting file has the same name as the input file. Instead, the cited Waters uses the hash to determine that the incorrect file is not in the memory and returned to the GET request.

For the above reasons, Applicants submit that claims 1, 10, and 19 are patentable over the cited art because the cited Waters does not disclose all the claim requirements.

The Examiner rejected claims 2-6, 11-15, 20-24, and 28-36 as obvious over Waters. Applicants submit that these claims are patentable over the cited Waters because they depend from one of amended claims 1, 10, and 19, respectively, which are patentable over the cited combination for the reasons described above. Further claims discussed below provide additional grounds of patentability over the cited art.

Claims 4, 13, and 22 depend from claims 3, 12, and 13, which require that the data structure includes an entry for each possible integer value capable of being generated from the hash function. Claims 4, 13, and 22 further require that processing the data structure to determine whether there is a preexisting file having the same name comprises determining whether the entry for the integer value to which the input file name maps indicates the presence of one preexisting file mapping to the same integer value as the input file name.

The Examiner cited col. 8, lines 15-21 of Waters as teaching the additional requirements of these claims. (Fifth Office Action, pg. 6) Applicants traverse.

The cited col. 8 mentions that after determining the hash value for the file path and name, the hash value is used as an index int the external memory and file at the offset in the address memory is retrieved to return to the GET request to the URI.

Nowhere does the cited col. 8 anywhere teach or suggest a data structure that is processed to determine whether there is a preexisting file for the hash value that has the same name. Instead, the cited Waters uses the hash as an index to return a file in the memory, not to determine whether there is a preexisting file having the same name as an input file as claimed. Further, the cited Waters does not provide a separate data structure to which the hash values match. Instead, in the cited Waters, the hash values comprise offsets into memory where the requested file is lcoated.

For these reasons, claims 4, 13, and 22 provide additional grounds of patentability over the cited combined art.

Claims 5, 14, and 23 depend from claims 4, 13, and 22 and further require that the data structure is a one-dimensional array and wherein each entry is capable of having one of two values, further comprising setting the entry to a first value if there is one preexisting file name in the file system that maps to the integer value for the entry, and wherein determining whether there is one preexisting file comprises determining whether the entry for the integer value to which the input file name maps has the first value.

The Examiner cited col. 8, lines 1-15 of Waters as teaching the additional requirements of these claims. (Fifth Office Action, pg. 6) Applicants traverse.

The cited col. 8 discusses how when the file path and name are received, the hash value is determined. The hash value provides an index into the memory, or an offset to the start of the actual file in the memory. (Waters, col. 8, lines 11-20).

Nowhere does the cited col. 8 anywhere disclose a data structure where each entry in the data structure mapping to the hash value has one of two values, where the first value is set if there is one preexisting file name that maps to the integer value for the entry, and determining whether there is a preexisting file with the same name by determining whether the entry for the integer value has the first value.

Nowhere does Waters teach or suggest setting a value in a hash table depending on whether there is a preexisting file having the name hashing to the value. Instead, the hash value of Waters is used as an offset into the memory where the file may be accessed. The cited Waters

does not provide a value in a table corresponding to the hash table indicating whether there is a preexisting file in the file system that maps to a value corresponding to an entry in the table. Instead, in the the cited Waters, the hash value comprises an offset into a memory device, not an index to a table indicating whether there is one preexisting file mapping to the integer value of the entry as claimed.

For these reasons, claims 5, 14, and 23 provide additional grounds of patentability over the cited combined art.

Claims 6, 15, and 24 depend from claims 1, 10, and 19 and further require applying the function to each file name in the file system to map each file name to one value and indicating in the data structure, for each file name, that there is one preexisting file for the value to which the file name maps.

The Examiner cited col. 8, lines 1-15 of Waters as teaching the additional requirements of these claims. (Office Action, pg. 6) Applicants traverse.

As discussed, the cited col. 8 mentions how when the file path and name are received, the hash value is determined. The hash value provides an index into the memory, or an offset to the start of the actual file in the memory. (Waters, col. 8, lines 11-20).

Thus, the hash is use to provide an offset into memory based on a file name. Nowhere does the cited Waters anywhere teach or suggest applying a function, or the hash of Waters, to each file name in a file system as claimed to indicate whether, for each file name, there is one preexisting file. Instead, as discussed, the cited Waters only discusses using the hash value as an offset into memory where the requested file is located.

For these reasons, the cited references do not teach or suggest, alone or in combination, the requirements of claims 6, 15, and 24.

Claims 28, 31, and 34 depend from claims 1, 10, and 19 and further require searching the file system for one preexisting file having the same name as the input file name if the data structure indicates that one preexisting file has a name that maps, according to the function, to the same value to which the input file maps; and performing an operation if the file system includes one preexisting file having the same name as the input file.

The Examiner cited FIG. 6 as teaching the additional requirements of these claims. (Office Action, pgs. 6-7). Applicants traverse.

The cited FIG. 6 discusses applying a hash function to a file path and name to determine a hash value providing an offset into an external memory where the file may be accessed. Nowhere does the cited FIG. 6 anywhere teach, suggest or mention searching the file system of the web server of Waters for one preexisting file having the same name as the input file name if the data structure indicates that one preexisting file has a name that maps, according to the function, to the same value to which the input file maps. Instead, the cited Waters uses the hash value as an offset into memory to access the file, not to determine whether there is a preexisting file and then searching the file system for the preexisting file.

Further, nowhere does the cited Waters anywhere disclose performing an operation if the file system includes one preexisting file having the same name as the input file.

For these reasons, the cited references do not teach or suggest, alone or in combination, the requirements of claims 28, 31, and 34.

Independent claims 41, 43, and 45 substantially include the requirements of claims 1, 10, and 19 and additionally require that the data structure includes multiple columns for different directories in the file system to indicate file names in different directories of the file system.

Applicants submit that these claims are patentable over the cited art for the reasons discussed with respect to claims 1, 10, and 19 because claims 41, 43, and 45 substantially include the requirements of claims 1, 10, and 19.

The Examiner found that Waters does not teach the claim requirement that the data structure has multiple columns for the different directories in the file system, but cited col. 7, lines 25-29 as teaching that a typical file has a file path. (Fifth Office Action, pg. 10). Applicants traverse.

The cited Waters does not teach or suggest a multi-column table for the different directories in the file system. Instead, the cited Waters applies a hash function to the file path name, which may include a directory path, to determine an offset into the memory where the requested file is located. Waters does not teach or suggest a data structure indicating values

corresponding to file names, but instead that the hash maps to an entry in a memory. Nowhere does the cited Waters anywhere teach that the memory has multiple columns for different directories in the file system. Instead, the memory of Waters includes the actual files being requested.

For these reasons, the cited references do not teach or suggest, alone or in combination, the requirements of claims 41, 43, and 45.

2. Claims 7-9, 16-18, and 25-27 are Patentable Over the Cited Art

The Examiner rejected claims 7-9, 16-18, and 25-27 as obvious (35 U.S.C. §103) over Waters in view of the Background section of the Application. (Office Action, pgs 10-11). Applicants traverse.

Claims 7-9, 16-18, and 25-27 are patentable over the cited art because they depend from one of claims 1, 10, and 19, which are patentable over the cited art for the reasons discussed above. Moreover, the requirements of these dependent claims in combination with the base claim requirements provide further grounds of patentability over the cited art.

3. Claims 37-39 are Patentable Over the Cited Art

The Examiner rejected claims 37-39 as obvious over Waters in view of Williams (U.S. Patent No. 5,990,810). Applicants traverse for the following reasons.

Applicants submit that added claims 37-39 are patentable over the cited art because they depend from one of claims 1, 10, and 19, which are patentable over the cited art for the reasons described above, and because the additional dependent limitations provide further grounds of patentability over the cited art. The Examiner cited col. 11, lines 25-45 of Williams as teaching the additional requirements of these claims. (Fifth Office Action, pg. 12) Applicants traverse for the following reasons.

The cited col. 11 of Williams discusses wide hash functions whose output values are wider, such that the probability of any two randomly chosen blocks having the same hashed value is negligible.

Although the cited Williams discusses wide hashes in general, nowhere does the cited Williams anywhere teach or suggest that a wide has is used to minimize the likelihood that two file names would be hashed to a same hash value. Nowhere does the cited Williams anywhere teach or suggest applying a wide hash to all the file names of a file system to avoid the likelihood that two file names would hash to a same value.

Accordingly, claims 37, 38, and 39 provide additional grounds of patentability over the cited art.

Conclusion

For all the above reasons, Applicant submits that the pending claims pending claims 1-39, 41, 43, and 45are patentable over the art of record. Applicants submit that no additional fees are needed. Nonetheless, should any additional fees be required, please charge Deposit Account No. 50-0585.

The attorney of record invites the Examiner to contact him at (310) 553-7977 if the Examiner believes such contact would advance the prosecution of the case.

Dated: September 17, 2004

By:

David W. Victor
Reg. No.: 39,867

Please direct all correspondences to:

David Victor
Konrad Raynes & Victor, LLP
315 South Beverly Drive, Ste. 210
Beverly Hills, CA 90212
Tel: 310-553-7977
Fax: 310-556-7984